Package ‘hypeR’

March 11, 2025

Title An R Package For Geneset Enrichment Workflows

Version 2.5.0

Description An R Package for Geneset Enrichment Workflows.

Depends R (>= 3.6.0)

License GPL-3 + file LICENSE

URL https://github.com/montilab/hypeR

Encoding UTF-8

RoxygenNote 7.2.3

LazyData false

Imports ggplot2, ggforce, R6, magrittr, dplyr, purrr, stats, stringr, scales, rlang, httr, openxlsx, htmltools, reshape2, reactable, msigdbr, kableExtra, rmarkdown, igraph, visNetwork, shiny, BiocStyle

Suggests tidyverse, devtools, testthat, knitr

biocViews GeneSetEnrichment, Annotation, Pathways

VignetteBuilder knitr

BugReports https://github.com/montilab/hypeR/issues

git_url https://git.bioconductor.org/packages/hypeR

git_branch devel

git_last_commit 2152271

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-03-11

Author Anthony Federico [aut, cre],
Andrew Chen [aut],
Stefano Monti [aut]

Maintainer Anthony Federico <anfed@bu.edu>

Contents

`.check_overlap` *Check overlap of signature across genesets*

Description

Check overlap of signature across genesets

Usage

```
.check_overlap(signature, genesets, threshold = 0.05)
```

Arguments

`signature`	A vector of symbols
`genesets`	A list of genesets
`threshold`	Minimum percent overlap

Value

Percent overlap

.dots_multi_plot *Plot top enriched genesets across multiple signatures*

Description

Plot top enriched genesets across multiple signatures

Usage

```
.dots_multi_plot(
  multihyp_data,
  top = 20,
  abrv = 50,
  size_by = c("genesets", "significance", "none"),
  pval_cutoff = 1,
  fdr_cutoff = 1,
  val = c("fdr", "pval"),
  title = ""
)
```

Arguments

multihyp_data	A list of hyp objects
top	Limit number of genesets shown
abrv	Abbreviation length of genesetlabels
size_by	Size dots by e.g. c("genesets", "significance", "none")
pval_cutoff	Filter results to be less than pval cutoff
fdr_cutoff	Filter results to be less than fdr cutoff
val	Choose significance value e.g. c("fdr", "pval")
title	Plot title

Value

A ggplot object

`.dots_plot` *Plot top enriched genesets*

Description

Plot top enriched genesets

Usage

```
.dots_plot(
  hyp_df,
  top = 20,
  abrv = 50,
  size_by = c("genesets", "significance", "none"),
  pval_cutoff = 1,
  fdr_cutoff = 1,
  val = c("fdr", "pval"),
  title = ""
)
```

Arguments

`hyp_df`	A dataframe from a hyp object
`top`	Limit number of genesets shown
`abrv`	Abbreviation length of genesetlabels
`size_by`	Size dots by e.g. c("genesets", "significance", "none")
`pval_cutoff`	Filter results to be less than pval cutoff
`fdr_cutoff`	Filter results to be less than fdr cutoff
`val`	Choose significance value e.g. c("fdr", "pval")
`title`	Plot title

Value

A ggplot object

`.enrichment_map` *Plot enrichment map*

Description

Plot enrichment map

Usage

```
.enrichment_map(
  hyp_df,
  genesets,
  similarity_metric = c("jaccard_similarity", "overlap_similarity"),
  similarity_cutoff = 0.2,
  pval_cutoff = 1,
  fdr_cutoff = 1,
  val = c("fdr", "pval"),
  top = NULL,
  title = ""
)
```

Arguments

`hyp_df` A dataframe from a hyp object

`genesets` A list of genesets

`similarity_metric` Metric to calculate geneset similarity

`similarity_cutoff` Geneset similarity cutoff

`pval_cutoff` Filter results to be less than pval cutoff

`fdr_cutoff` Filter results to be less than fdr cutoff

`val` Choose significance value shown above nodes e.g. c("fdr", "pval")

`top` Limit number of pathways shown

`title` Plot title

Value

A visNetwork object

`.find_members` *Find geneset members*

Description

Find geneset members

Usage

```
.find_members(id, genesets, nodes, edges)
```

Arguments

`id`	A vector of ids
`genesets`	A list of genesets (see `rgsets`)
`nodes`	A data frame of labeled nodes (see `rgsets`)
`edges`	A data frame of directed edges (see `rgsets`)

Value

A vector of ids

`.format_str` *Format a string using placeholders*

Description

Format a string using placeholders

Usage

```
.format_str(string, ...)
```

Arguments

`string`	A an unformatted string with placeholders
`...`	Variables to format placeholders with

Value

A formatted string

Examples

```
## Not run:
format_str("Format with {1} and {2}", "x", "y")

## End(Not run)
```

.github_rds *Load an rds file directly from github*

Description

Load an rds file directly from github

Usage

```
.github_rds(url)
```

Arguments

url	A url

Value

A list

.hexa *Adjust alpha of a hex string*

Description

Adjust alpha of a hex string

Usage

```
.hexa(hex, percent = 1)
```

Arguments

hex	A 6-character hex string (e.g. #000000)
percent	Alpha level from 0-1

Value

A hex string

.hiearchy_map *Plot hiearchy map*

Description

Plot hiearchy map

Usage

```
.hiearchy_map(
  hyp_df,
  rgsets_obj,
  pval_cutoff = 1,
  fdr_cutoff = 1,
  val = c("fdr", "pval"),
  top = NULL,
  title = "",
  graph = FALSE
)
```

Arguments

hyp_df	A dataframe from a hyp object
rgsets_obj	A relatonal geneset from a hyp object
pval_cutoff	Filter results to be less than pval cutoff
fdr_cutoff	Filter results to be less than fdr cutoff
val	Choose significance value displayed when hovering nodes e.g. c("fdr", "pval")
top	Limit number of pathways shown
title	Plot title
graph	Return an igraph object instead

Value

A visNetwork object

`.hyperdb_url` *Get base url for hyperdb*

Description

Get base url for hyperdb

Usage

```
.hyperdb_url(api = FALSE)
```

Value

A base url

`.hyper_enrichment` *Overrepresentation test via hyper-geometric distribution*

Description

Overrepresentation test via hyper-geometric distribution

Usage

```
.hyper_enrichment(
  signature,
  genesets,
  background = length(unique(unlist(genesets))),
  plotting = TRUE
)
```

Arguments

`signature`	A vector of symbols
`genesets`	A list of genesets
`background`	Size of background population genes
`plotting`	Use true to generate plots

Value

A list of data and plots

`.jaccard_similarity` *Calculate jaccard similarity of two sets*

Description

Calculate jaccard similarity of two sets

Usage

```
.jaccard_similarity(a, b)
```

Arguments

a	A vector
b	A vector

Value

A numerical value

`.kstest` *One-sided Kolmogorov–Smirnov test*

Description

One-sided Kolmogorov–Smirnov test

Usage

```
.kstest(
  n.x,
  y,
  weights = NULL,
  weights.pwr = 1,
  absolute = FALSE,
  plotting = FALSE,
  plot.title = ""
)
```

Arguments

`n.x`	The length of a ranked list
`y`	A vector of positions in the ranked list
`weights`	Weights for weighted score (Subramanian et al.)
`weights.pwr`	Exponent for weights (Subramanian et al.)
`absolute`	Takes max-min score rather than the max deviation from null
`plotting`	Use true to generate plot
`plot.title`	Plot title

Value

A list of data and plots

`.ks_enrichment` *Enrichment test via one-sided Kolmogorov–Smirnov test*

Description

Enrichment test via one-sided Kolmogorov–Smirnov test

Usage

```
.ks_enrichment(
  signature,
  genesets,
  weights = NULL,
  weights.pwr = 1,
  absolute = FALSE,
  plotting = TRUE
)
```

Arguments

`signature`	A vector of ranked symbols
`genesets`	A list of genesets
`weights`	Weights for weighted score (Subramanian et al.)
`weights.pwr`	Exponent for weights (Subramanian et al.)
`absolute`	Takes max-min score rather than the max deviation from null
`plotting`	Use true to generate plot

Value

A list of data and plots

`.overlap_similarity` *Calculate overlap similarity of two sets*

Description

Calculate overlap similarity of two sets

Usage

```
.overlap_similarity(a, b)
```

Arguments

`a`	A vector
`b`	A vector

Value

A numerical value

`.reverselog_trans` *Custom reverse log transformation of continous ggplot axes*

Description

Custom reverse log transformation of continous ggplot axes

Usage

```
.reverselog_trans(base = exp(1))
```

Arguments

`base`	Logarithm base

`.string_args` *Convert an arguments list to string format*

Description

Convert an arguments list to string format

Usage

```
.string_args(args)
```

Arguments

args	A list of keyword arguments

Value

A string of keyword arguments

Examples

```
## Not run:
string_args(list(x=15, y="fdr", z=TRUE))

## End(Not run)
```

`clean_genesets` *Clean labels of genesets*

Description

Clean labels of genesets

Usage

```
clean_genesets(x)
```

Arguments

x	A vector of labels

Examples

```
HALLMARK <- msigdb_download("Homo sapiens", "H", "")
names(HALLMARK) <- clean_genesets(names(HALLMARK))
head(names(HALLMARK))
```

enrichr_available *Get enrichr available genesets*

Description

Get enrichr available genesets

Usage

```
enrichr_available(
  db = c("Enrichr", "YeastEnrichr", "FlyEnrichr", "WormEnrichr", "FishEnrichr")
)
```

Arguments

db	A species

Value

A dataframe of available genesets

Examples

```
enrichr_available()
```

enrichr_connect *Connect to the enrichr web application*

Description

Connect to the enrichr web application

Usage

```
enrichr_connect(
  endpoint,
  db = c("Enrichr", "YeastEnrichr", "FlyEnrichr", "WormEnrichr", "FishEnrichr")
)
```

Arguments

endpoint	The url endpoint to connect to
db	A species

Value

A web response

enrichr_download *Download data from enrichr in the form of a named list*

Description

Download data from enrichr in the form of a named list

Usage

```
enrichr_download(
  genesets,
  db = c("Enrichr", "YeastEnrichr", "FlyEnrichr", "WormEnrichr", "FishEnrichr")
)
```

Arguments

genesets	A name corresponding to available genesets
db	A species

Value

A list of genesets

Examples

```
ATLAS <- enrichr_download("Human_Gene_Atlas")
```

enrichr_gsets *Download data from enrichr in the form of a gsets object*

Description

Download data from enrichr in the form of a gsets object

Usage

```
enrichr_gsets(
  genesets,
  db = c("Enrichr", "YeastEnrichr", "FlyEnrichr", "WormEnrichr", "FishEnrichr"),
  clean = FALSE
)
```

Arguments

`genesets`	A name corresponding to available genesets
`db`	A species
`clean`	Use true to clean labels of genesets

Value

A gsets object

Examples

```
ATLAS <- enrichr_gsets("Human_Gene_Atlas")
```

`enrichr_urls` *Get url base for species-specific enrichr libraries*

Description

Get url base for species-specific enrichr libraries

Usage

```
enrichr_urls(
  db = c("Enrichr", "YeastEnrichr", "FlyEnrichr", "WormEnrichr", "FishEnrichr")
)
```

Arguments

`db`	A species

Value

A url

genesets_Server *Shiny server module for geneset selection*

Description

Shiny server module for geneset selection

Usage

```
genesets_Server(id, clean = FALSE)
```

Arguments

id	A unique namespace identifier matching to interface
clean	Use true to clean geneset names

Value

Shiny server code

genesets_UI *Shiny interface module for geneset selection*

Description

Shiny interface module for geneset selection

Usage

```
genesets_UI(id)
```

Arguments

id	A unique namespace identifier

Value

Shiny ui elements

`ggempty` *An empty ggplot*

Description

An empty ggplot

Usage

```
ggempty()
```

Value

A ggplot object

`ggeplot` *Enrichment plot implemented in ggplot*

Description

Enrichment plot implemented in ggplot

Usage

```
ggeplot(n, positions, x_axis, y_axis, title = "")
```

Arguments

`n`	The length of a ranked list
`positions`	A vector of positions in the ranked list
`x_axis`	The x-axis of a running enrichment score
`y_axis`	The y-axis of a running enrichment score
`title`	Plot title

Value

A ggplot object

`ggvenn` *Venn diagram implemented in ggplot*

Description

Venn diagram implemented in ggplot

Usage

```
ggvenn(a, b, ga, gb, title = "")
```

Arguments

`a`	A vector for group a
`b`	A vector for group b
`ga`	A string label for group a
`gb`	A string label for group b
`title`	Plot title

Value

A ggplot object

`gsets` *A genesets object*

Description

A genesets object

A genesets object

See Also

`rgsets`

Public fields

`genesets` A named list of genesets

`name` A character vector describing source of genesets

`version` A character vector describing versioning

Methods

Public methods:

- `gsets$new()`
- `gsets$print()`
- `gsets$list()`
- `gsets$info()`
- `gsets$reduce()`
- `gsets$clone()`

Method `new()`**:** Create a gsets object

Usage:

```
gsets$new(
  genesets,
  name = "Custom",
  version = "",
  clean = FALSE,
  quiet = FALSE
)
```

Arguments:

`genesets` A named list of genesets

`name` A character vector describing source of genesets

`version` A character vector describing versioning

`clean` Use true to clean labels of genesets

`quiet` Use true to silence warnings

Returns: A new gsets object

Method `print()`**:** Print genesets information

Usage:

```
gsets$print()
```

Returns: NULL

Method `list()`**:** Return genesets as a list

Usage:

```
gsets$list()
```

Returns: A list of genesets

Method `info()`**:** Returns versioning information

Usage:

```
gsets$info()
```

Returns: A character vector with name and version

Method `reduce()`**:** Reduces genesets to a background distribution of symbols

Usage:

`gsets$reduce(background)`

Arguments:

`background` A character vector of symbols

Returns: A gsets object

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

`gsets$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Examples

```
genesets <- list("GSET1" = c("GENE1", "GENE2", "GENE3"),
                 "GSET2" = c("GENE4", "GENE5", "GENE6"),
                 "GSET3" = c("GENE7", "GENE8", "GENE9"))

gsets_obj <- gsets$new(genesets, name="example", version="v1.0")
print(gsets_obj)
```

`hyp` *A hyp object*

Description

A hyp object

A hyp object

See Also

`multihyp`

Public fields

`data` A dataframe returned by hypeR()

`plots` A list of plots returned by hypeR()

`args` A list of arguments passed to hypeR()

`info` Exported information for reproducibility

Methods

Public methods:

- `hyp$new()`
- `hyp$print()`
- `hyp$as.data.frame()`
- `hyp$clone()`

Method `new()`**:** Create a hyp object

Usage:

```
hyp$new(data, plots = NULL, args = NULL, info = NULL)
```

Arguments:

`data` A dataframe returned by hypeR()

`plots` A list of plots returned by hypeR()

`args` A list of arguments passed to hypeR()

`info` Exported information for reproducibility

Returns: A new hyp object

Method `print()`**:** Print hyp obect

Usage:

```
hyp$print()
```

Returns: NULL

Method `as.data.frame()`**:** Extract dataframe from hyp obect

Usage:

```
hyp$as.data.frame()
```

Returns: NULL

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
hyp$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
data <- data.frame(replicate(5,sample(0:1,10,rep=TRUE)))
args <- list("arg_1"=1, "arg_2"=2, "arg_3"=3)
hyp_obj <- hyp$new(data, args=args)
```

hypeR *Calculate enrichment of one or more signatures*

Description

Calculate enrichment of one or more signatures

Usage

```
hypeR(
  signature,
  genesets,
  test = c("hypergeometric", "kstest"),
  background = 23467,
  power = 1,
  absolute = FALSE,
  pval = 1,
  fdr = 1,
  plotting = FALSE,
  quiet = TRUE
)
```

Arguments

`signature`	A vector of symbols
`genesets`	A gsets/rgsets object or a named list of genesets
`test`	Choose an enrichment type e.g. c("hypergeometric", "kstest")
`background`	Size or character vector of background population genes
`power`	Exponent for weights (kstest only)
`absolute`	Takes max-min score rather than the max deviation from null (kstest only)
`pval`	Filter results to be less than pval cutoff
`fdr`	Filter results to be less than fdr cutoff
`plotting`	Use true to generate plots for each geneset test (may slow performance)
`quiet`	Use true to suppress logs and warnings

Value

A hyp object

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)
```

---

hyperdb_available *Check available data to download from hyperdb*

Description

Check available data to download from hyperdb

Usage

```
hyperdb_available()
```

Examples

```
hyperdb_available()
```

---

hyperdb_gsets *Download data from hyperdb*

Description

Download data from hyperdb

Usage

```
hyperdb_gsets(source, gsets)
```

Arguments

source	A source identifier
gsets	A genesets identifier

Value

A list

Examples

```
KEGG <- hyperdb_gsets("KEGG", "KEGG_v92.0.rds")
```

hyperdb_rgsets *Download data from hyperdb in the form of a rgsets object*

Description

Download data from hyperdb in the form of a rgsets object

Usage

```
hyperdb_rgsets(rgsets, version)
```

Arguments

rgsets	A name corresponding to an available relational genesets object
version	A version number

Value

An rgsets object

Examples

```
REACTOME <- hyperdb_rgsets("REACTOME", "70.0")
```

hyp_dots *Visualize hyp/multihyp objects as a dots plot*

Description

Visualize hyp/multihyp objects as a dots plot

Usage

```
hyp_dots(
  hyp_obj,
  top = 20,
  abrv = 50,
  size_by = c("genesets", "significance", "none"),
  pval = 1,
  fdr = 1,
  val = c("fdr", "pval"),
  title = "",
  merge = FALSE
)
```

Arguments

`hyp_obj`	A hyp or multihyp object
`top`	Limit number of genesets shown
`abrv`	Abbreviation length of geneset labels
`size_by`	Size dots by e.g. c("genesets", "significance", "none")
`pval`	Filter results to be less than pval cutoff
`fdr`	Filter results to be less than fdr cutoff
`val`	Choose significance value for plot e.g. c("fdr", "pval")
`title`	Plot title
`merge`	Use true to merge a multihyp object into one plot

Value

A ggplot object

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

hyp_dots(hyp_obj, val="fdr")
```

`hyp_emap` *Visualize hyp/multihyp objects as an enrichment map*

Description

Visualize hyp/multihyp objects as an enrichment map

Usage

```
hyp_emap(
  hyp_obj,
  similarity_metric = c("jaccard_similarity", "overlap_similarity"),
  similarity_cutoff = 0.2,
  pval = 1,
  fdr = 1,
  val = c("fdr", "pval"),
  top = NULL,
  title = ""
)
```

Arguments

hyp_obj	A hyp or multihyp object
similarity_metric	Metric to calculate geneset similarity
similarity_cutoff	Geneset similarity cutoff
pval	Filter results to be less than pval cutoff
fdr	Filter results to be less than fdr cutoff
val	Choose significance value shown above nodes e.g. c("fdr", "pval")
top	Limit number of pathways shown
title	Plot title

Value

A visNetwork object

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

hyp_emap(hyp_obj, top=30, val="fdr")
```

---

hyp_hmap *Visualize hyp/multihyp objects as a hiearchy map*

Description

Visualize hyp/multihyp objects as a hiearchy map

Usage

```
hyp_hmap(
  hyp_obj,
  pval = 1,
  fdr = 1,
  val = c("fdr", "pval"),
  top = NULL,
  title = "",
  graph = FALSE
)
```

Arguments

`hyp_obj`	A hyp or multihyp object
`pval`	Filter results to be less than pval cutoff
`fdr`	Filter results to be less than fdr cutoff
`val`	Choose significance value displayed when hovering nodes e.g. c("fdr", "pval")
`top`	Limit number of pathways shown
`title`	Plot title
`graph`	Return an igraph object instead

Value

A visNetwork object

Examples

```
genesets <- hyperdb_rgsets("REACTOME", "70.0")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

hyp_hmap(hyp_obj, top=60)
```

`hyp_show` *Convert a hyp object to a reactable table*

Description

Convert a hyp object to a reactable table

Usage

```
hyp_show(hyp_obj, simple = FALSE)
```

Arguments

`hyp_obj`	A hyp object
`simple`	Use true to only include essential columns

Value

A reactable table

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

hyp_show(hyp_obj)
```

hyp_to_excel *Export hyp/multihyp object to excel*

Description

Export hyp/multihyp object to excel

Usage

```
hyp_to_excel(hyp_obj, file_path, cols = NULL, versioning = TRUE)
```

Arguments

hyp_obj	A hyp or multihyp object
file_path	A file path
cols	Dataframe columns to include
versioning	Add sheet with versioning information

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

hyp_to_excel(hyp_obj, file_path="pathways.xlsx")
```

hyp_to_graph *Convert a hyp object to an igraph object*

Description

Convert a hyp object to an igraph object

Usage

```
hyp_to_graph(hyp_obj)
```

Arguments

hyp_obj	A hyp object

Value

An igraph object

Examples

```
genesets <- hyperdb_rgsets("REACTOME", "70.0")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

ig <- hyp_to_graph(hyp_obj)
```

hyp_to_rmd *Export hyp object to rmarkdown*

Description

Export hyp object to rmarkdown

Usage

```
hyp_to_rmd(
  hyp_obj,
  file_path,
  title = "Workflow Report",
  subtitle = "",
  author = "",
  header = "Results",
  versioning = TRUE,
  show_dots = TRUE,
  show_emaps = TRUE,
  show_hmaps = FALSE,
  show_tables = TRUE,
  hyp_dots_args = list(top = 15, val = "fdr"),
  hyp_emap_args = list(top = 25, val = "fdr", similarity_metric = "jaccard_similarity",
    similarity_cutoff = 0.2),
  hyp_hmap_args = list(top = 25, val = "fdr"),
  custom_rmd_config = NULL,
  custom_pre_content = NULL,
  custom_post_content = NULL,
  session_info = FALSE
)
```

Arguments

`hyp_obj`	A hyp object, multihyp object, or list of multihyp objects
`file_path`	A file path
`title`	Title of markdown report
`subtitle`	Subtitle of markdown report
`author`	Authors of markdown report
`header`	Header name of tabset section
`versioning`	Add versioning information
`show_dots`	Option to show dots plots in tabs
`show_emaps`	Option to show enrichment maps in tabs
`show_hmaps`	Option to show hiearchy maps in tabs
`show_tables`	Option to show table in tabs
`hyp_dots_args`	A list of keyword arguments passed to hyp_dots
`hyp_emap_args`	A list of keyword arguments passed to hyp_emap
`hyp_hmap_args`	A list of keyword arguments passed to hyp_hmap
`custom_rmd_config`	Replace configuration section of markdown report
`custom_pre_content`	Insert custom content before tabset section

`custom_post_content`	Insert custom content after tabset section
`session_info`	Use true to include session info

hyp_to_table *Export hyp/multihyp object to table*

Description

Export hyp/multihyp object to table

Usage

```
hyp_to_table(hyp_obj, file_path, sep = "\t", cols = NULL, versioning = TRUE)
```

Arguments

`hyp_obj`	A hyp or multihyp object
`file_path`	A file path for hyp objects and directory for multihyp objects
`sep`	The field separator string
`cols`	Dataframe columns to include
`versioning`	Add header with versioning information

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

hyp_to_table(hyp_obj, file_path="pathways.txt")
```

`limma` *Differential Expression*

Description

A differential expression table

Usage

```
limma
```

Format

A data frame

`msigdb_available` *Get msigdbr available genesets*

Description

Get msigdbr available genesets

Usage

```
msigdb_available(species = "Homo sapiens")
```

Arguments

`species`	A species to determine gene symbols (refer to ?msigdbr::msigdbr for avilable species)

Value

A dataframe of available genesets

Examples

```
msigdb_available("Homo sapiens")
```

`msigdb_check_species` *Check if species is available*

Description

Check if species is available

Usage

```
msigdb_check_species(species = "")
```

Arguments

`species`	A species

Examples

```
## Not run:
msigdb_check_species("Homo sapiens")

## End(Not run)
```

`msigdb_download` *Download data from msigdb in the form of a named list*

Description

Download data from msigdb in the form of a named list

Usage

```
msigdb_download(species, category, subcategory = "")
```

Arguments

`species`	A species to determine gene symbols (refer to ?msigdbr::msigdbr for avilable species)
`category`	Geneset category (refer to ?msigdbr::msigdbr for avilable categories)
`subcategory`	Geneset subcategory (refer to ?msigdbr::msigdbr for avilable subcategories)

Value

A list of genesets

Examples

```
HALLMARK <- msigdb_download("Homo sapiens", "H", "")
```

---

msigdb_gsets *Download data from msigdb in the form of a gsets object*

Description

Download data from msigdb in the form of a gsets object

Usage

```
msigdb_gsets(species, category, subcategory = "", clean = FALSE)
```

Arguments

species	A species to determine gene symbols (refer to ?msigdbr::msigdbr for avilable species)
category	Geneset category (refer to ?msigdbr::msigdbr for avilable categories)
subcategory	Geneset subcategory (refer to ?msigdbr::msigdbr for avilable subcategories)
clean	Use true to clean labels of genesets

Value

A gsets object

Examples

```
HALLMARK <- msigdb_gsets("Homo sapiens", "H", "")
```

---

msigdb_info *Print msigdb gsets information*

Description

Print msigdb gsets information

Usage

```
msigdb_info()
```

Examples

```
msigdb_info()
```

`msigdb_species` *Get msigdbr available species*

Description

Get msigdbr available species

Usage

```
msigdb_species()
```

Value

A character vector of species

Examples

```
msigdb_species()
```

`msigdb_version` *Get msigdbr package version number*

Description

Get msigdbr package version number

Usage

```
msigdb_version()
```

Value

Version number

Examples

```
msigdb_version()
```

multihyp *A multihyp object*

Description

A multihyp object

A multihyp object

See Also

hyp

Public fields

data A list of hyp objects

Methods

Public methods:

- multihyp$new()
- multihyp$print()
- multihyp$as.list()
- multihyp$clone()

Method new()**:** Create a multihyp object

Usage:

```
multihyp$new(data)
```

Arguments:

data A list of hyp objects

Returns: A new multihyp object

Method print()**:** Print multihyp obect

Usage:

```
multihyp$print()
```

Returns: NULL

Method as.list()**:** Print multihyp obect

Usage:

```
multihyp$as.list()
```

Returns: A list of hyp objects as dataframes

Method clone()**:** The objects of this class are cloneable with this method.

Usage:

```
multihyp$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
data <- data.frame(replicate(5,sample(0:1,10,rep=TRUE)))
args <- list("arg_1"=1, "arg_2"=2, "arg_3"=3)
hyp_obj <- hyp$new(data, args=args)
data <- list("hyp_1"=hyp_obj, "hyp_2"=hyp_obj,"hyp_3"=hyp_obj)
multihyp_obj <- multihyp$new(data)
```

---

pvector *A push/pop capable vector*

Description

A push/pop capable vector

A push/pop capable vector

Public fields

`values` A vector of values

Methods

Public methods:

- `pvector$new()`
- `pvector$print()`
- `pvector$length()`
- `pvector$pop()`
- `pvector$push()`
- `pvector$clone()`

Method `new()`**:** Create a pvector

Usage:

```
pvector$new(values = c())
```

Arguments:

`values` A vector of values

Returns: A new pvector

Method `print()`**:** Print pvector

Usage:

```
pvector$print()
```

Returns: NULL

Method `length()`**:** Get length of pvector

Usage:

```
pvector$length()
```

Returns: An integer

Method `pop()`**:** Pop vector

Usage:

```
pvector$pop()
```

Returns: Popped value

Method `push()`**:** Push values

Usage:

```
pvector$push(pushed.values)
```

Arguments:

`pushed.values` A vector of values

Returns: NULL

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
pvector$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`rctbl_build` *Reactable builder for hyp or mhyp objects*

Description

Reactable builder for hyp or mhyp objects

Usage

```
rctbl_build(obj, ...)
```

Arguments

`obj`	A hyp or multihyp object
`...`	Arguments passed to table generators

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")$genesets[1:5]

experiment <- list("S1"=c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502"),
                   "S2"=c("PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC"))

mhyp_obj <- hypeR(experiment, genesets, background=2522)

rctbl_build(mhyp_obj)
```

rctbl_hyp *Reactable table for hyp objects*

Description

Reactable table for hyp objects

Usage

```
rctbl_hyp(
  hyp,
  type = c("inner", "outer"),
  show_emaps = FALSE,
  show_hmaps = FALSE,
  hyp_emap_args = list(top = 25, val = "fdr"),
  hyp_hmap_args = list(top = 25, val = "fdr")
)
```

Arguments

hyp	A hyp object
type	Use style class for outer or inner tables
show_emaps	Option to show enrichment maps in tabs
show_hmaps	Option to show hiearchy maps in tabs
hyp_emap_args	A list of keyword arguments passed to hyp_emap
hyp_hmap_args	A list of keyword arguments passed to hyp_hmap

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")$genesets[1:5]

signature <- c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502",
               "PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC",
               "IDH2","IDH1","OGDHL","PC","SDHA","SUCLG1","SUCLA2","SUCLG2")

hyp_obj <- hypeR(signature, genesets, background=2522)

rctbl_hyp(hyp_obj)
```

rctbl_mhyp *Reactable table for multihyp objects*

Description

Reactable table for multihyp objects

Usage

```
rctbl_mhyp(
  mhyp,
  show_emaps = FALSE,
  show_hmaps = FALSE,
  hyp_emap_args = list(top = 25, val = "fdr"),
  hyp_hmap_args = list(top = 25, val = "fdr")
)
```

Arguments

mhyp	A multihyp object
show_emaps	Option to show enrichment maps in tabs
show_hmaps	Option to show hiearchy maps in tabs
hyp_emap_args	A list of keyword arguments passed to hyp_emap
hyp_hmap_args	A list of keyword arguments passed to hyp_hmap

Examples

```
genesets <- msigdb_gsets("Homo sapiens", "C2", "CP:KEGG")$genesets[1:5]

experiment <- list("S1"=c("IDH3B","DLST","PCK2","CS","PDHB","PCK1","PDHA1","LOC642502"),
                   "S2"=c("PDHA2","LOC283398","FH","SDHD","OGDH","SDHB","IDH3A","SDHC"))

mhyp_obj <- hypeR(experiment, genesets, background=2522)

rctbl_mhyp(mhyp_obj)
```

rgsets *A relational genesets object*

Description

A relational genesets object

A relational genesets object

See Also

gsets

Public fields

`genesets` A list of genesets where list names refers to geneset labels and values are geneset members represented as a vector

`nodes` A data frame of labeled nodes

`edges` A data frame of directed edges

`name` A character vector describing source of genesets

`version` A character vector describing versioning

Methods

Public methods:

- `rgsets$new()`
- `rgsets$print()`
- `rgsets$info()`
- `rgsets$reduce()`
- `rgsets$subset()`
- `rgsets$clone()`

Method `new()`**:** Create a rgsets object

Usage:

```
rgsets$new(
  genesets,
  nodes,
  edges,
  name = "Custom",
  version = "",
  quiet = FALSE
)
```

Arguments:

`genesets` A list of genesets where list names refers to geneset labels and values are geneset members represented as a vector

`nodes` A data frame of labeled nodes

`edges` A data frame of directed edges

`name` A character vector describing source of genesets

`version` A character vector describing versioning

`quiet` Use true to silence warnings

Returns: A new rgsets object

Method `print()`**:** Print relational genesets information

Usage:

```
rgsets$print()
```

Returns: NULL

Method `info()`: Returns versioning information

Usage:

```
rgsets$info()
```

Returns: A character vector with name and version

Method `reduce()`: Reduces genesets to a background distribution of symbols

Usage:

```
rgsets$reduce(background)
```

Arguments:

`background` A character vector of symbols

Returns: A rgsets object

Method `subset()`: Subsets genesets on a character vector of labels

Usage:

```
rgsets$subset(labels)
```

Arguments:

`labels` A character vector of genesets

Returns: A rgsets object

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
rgsets$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`wgcna` *Co-expression Modules*

Description

A list of co-expression modules

Usage

```
wgcna
```

Format

A nested list of character vectors

Index